International Journal of Computer Science Engineering
and Information Technology Research (IJCSEITR)
ISSN (P): 2249-6831; ISSN (E): 2249-7943
Vol. 7, Issue 4, Aug 2017, 31-36

EFFICIENT TRAFFIC MANAGEMENT WITH INTERMEDIATE DATA PARTITION FOR BIG DATA APPLICATIONS

NAVEEN WALI[1] & DR. SHASHIKUMAR D.R[2]

[1]*MTech, Computer Science and Engineering, Cambridge Institute of Technology, Bangalore, India*

[2]*Head, Computer Science and Engineering, Cambridge Institute of Technology, Bangalore, India*

ABSTRACT

The MapReduce programming models improves huge scale information handling on ware bunch by exploiting parallel map tasks and reduce tasks. Although many numerous endeavors have been made to enhance the execution of Map Reduce employments, they overlook the system movement produced in the shuffle stage, which assumes a basic part in execution improvement. Customarily, a hash function is used to parcel halfway information among lessen assignments, which, in any case, is not activity productive in fact that system topology and information estimate related with each key are not contemplated. In this paper, we study to lessen organize activity fetched for a Map Reduce work by outlining a novel middle of the road information segment conspire. Besides, we together consider the aggregator position issue, where every aggregator can decrease consolidated activity from various guide assignments. A decay based conveyed calculation is proposed to manage the vast scale enhancement issue for enormous information application and an online calculation is likewise intended to change information parcel and accumulation in a dynamic way. At last, broad recreation comes about exhibit that our recommendations can essentially lessen arrange activity taken a toll under both disconnected and online cases

KEYWORDS: ***Online Method, Distribution of Data& Hash Code***

Received: Jun 28, 2017; **Accepted:** Jul 15, 2017; **Published:** Jul 18, 2017; **Paper Id.:**IJCSEITRAUG20175

Original Article

INTRODUCTION

MapReduce framework has evolved as the most popular computing design for large data processing as it has less complex software developing model and automatically performs parallel execution. Map Reduce [1] [2] [3] implementation with Hadoop [4] [5] has been implemented by most MNC's organisations and companies, ie, Yahoo!, Google and Facebook, for various big data applications, like machine learning [6] [7] [8], bioinformatics [9] [10] [11], and cyber security [12] [13]. Computation phase mainly divides a data into phases, namely first phase which map and the second is reduce phase, which in turn are computed with several map tasks and then the secondary phase reduce tasks. In the phase of the map, the tasks in map are paralleled which converts the initial input splits into a form of key/value pairs which forms the intermediate phase. The key/value pairs thus formed are saved on the base machine are divided into multiple data partitions, and these are taken one per reduce task. Second phase, which is known as the reduce phase, the final data is produced by reduced tasks which allocates its own share of data parts from all map.

Earlier, there was intermediate data which is shuffled [14] [16] by using hash function [15] [17] step between reduced phase and map. Where, the data produced during first map phase undergoes several process doing ordering, partitioning and transferring data to the respective machines, which then works on reduce phase, respectively. This process results in a great volume of network traffic by the data produced during the two phases, slowing down the speed of data analytic applications.

RELATED WORK

Most existing work focuses on MapReduce performance improvement by optimizing its data transmission. Blanca et al. [18] have investigated the question of whether optimizing network usage can lead to better system performance and found that high network utilization and low network congestion shouldbe achieved simultaneously.

Palanisamy (a) Without global aggregation (b) With global aggregation et al. [19] have presented Purlieus, a MapReduce resource allocation system, to enhance the performance of MapReduce jobs in the cloud by locating intermediate data to the local machines or close-by physical machines. This locality-awareness reduces network traffic in the shuffle phase generated in the cloud data center. However, little work has studied to optimize network performance of the shuffle process that generates large amounts of data traffic in MapReduce jobs. A critical factor to the network performance in the shuffle phase is the intermediate data partition. The default scheme adopted by Hadoop is hash-based partition that would yield unbalanced loads among reduce tasks due to its unawareness of the data size associated with each key. To overcome this shortcoming, Ibrahim et al. [19] have developed a fairness-aware key partition approach that keeps track of the distribution of intermediate keys frequencies, and guarantees a fair distribution among reduced tasks. In addition to data partition, many efforts have been made on local aggregation, in-mapper combining and in-network aggregation to reduce network traffic within for a job with good performance.

METHODS

A decomposition-based distributed algorithm is proposed to deal with the large-scale optimization problem for big data application, and an online algorithm is also designed to adjust data partition and aggregation in a dynamic manner.

MAINPART

The framework has three cloud storages

- Top cloud storage.
- Region1 cloud storage.
- Region2 cloud storage.

For each region, there will be a web server which get the Request from the gateway server and fetch the data from their respective cloud storage and return back to gateway server. Gateway server is the main server, which has file details, Logical Block Address (LBA), IP lookup details and region server details. Once client request receives in gateway server, it has to identify the client IP belonging to the region, then it has to identify client requested file blocks through LBA.

It has to check in region server whether identified blocks are available or not, and instruct the main cloud storage to transfer unavailable block storage region server.

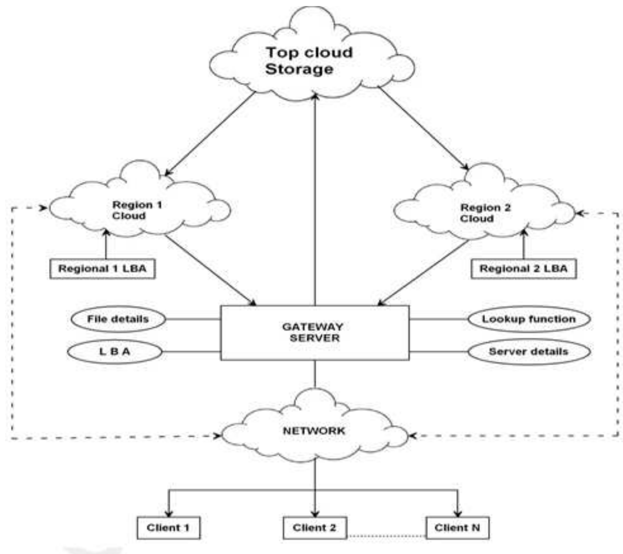

Figure 1: The Overall Block Diagram for Experimental Studies

Distributed Design Algorithm

We built up a distributed algorithm to take care of the issue on different machines in a parallel way. Our essential thought is to deteriorate the first huge scale issue into a few distributively resolvable sub issues, that are facilitated by an high-level master problem

- 1: set t = 1, and _p

  j (j ∈ A, p ∈ P) to arbitrary on negative values;
- 2: for t < T do
- 3: distributively solve the subproblem SUB DP and

  SUB AP on multiple machines in a parallel manner;
- 4: update the values of _pj with the gradient method

  and send the results to all subproblems;
- 5: set t = t + 1;
- 6:end for

Online Algorithm

In this distributed design, the data partition is distributed on multiple servers; we have to hold up all mappers to complete some time, recently beginning decreased undertakings, or direct estimation through profiling on a little arrangement of information. Practically speaking, mapreduce assignments may mostly cover in execution to increment framework throughput, and it is hard to gauge framework parameters at a high precision for huge information applications. These motivate us to plan an online algorithm to progressively alter information parcel and collection, amid the execution of map and reduce tasks.

SYSTEM DESIGN

Interface outline portrays the structure and association of the UI. It incorporates a portrayal of screen format, a meaning of the methods of cooperation and a depiction of route systems. Interface Control systems to execute route choices, the fashioner chooses frame one of various connection instruments. Interface Design work process starts with the ID of client, assignment, and ecological prerequisites as shown in Figure 1. When client errands have been distinguished, client situations are made and examined to characterize an arrangement of interface protests and activities.

Aesthetic outline, likewise, called graphic design portrays the "look and feel" of the WebApp. It incorporates shading plans and geometric design, content size, textual style and position, the utilization of representation and related tasteful choices. Content design-defines the format, structure, and framework for all substance that is exhibited as a major aspect of the WebApp. It builds up the connections between substance objects. Navigation plan speaks to the navigational stream between substance objects and for all WebApp capacities.

Engineering configuration distinguishes the general hypermedia structure for the WebApp. Engineering configuration is attached to the objectives build up for a WebApp, the substance to be exhibited, the clients who will visit, and the route rationality that has been set up, as shown in J2EE and MVC Architecture.

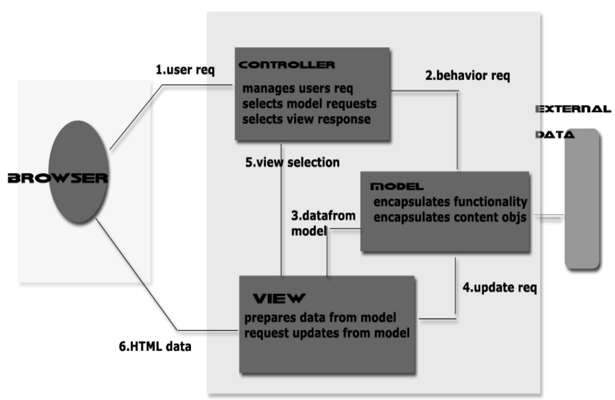

Figure 2: J2EE Uses MVC Architecture

RESULTS

We show the network traffic cost under different number of reduced tasks. We investigate the performance of these algorithms under different number of aggregators. We observe the online algorithm outperforms other two schemes. As the aggregator numbers increase, it is more beneficial to aggregate intermediate data and reducing the amount of data in the shuffle process. However, when the number of aggregators is set to 0, which means no global aggregation and our online algorithm mall ways achieves the lowest cost.

CONCLUSIONS

We displayed a novel way to deal with securing individual and business information in the Cloud. We propose observing information to get designs by profiling client conduct, to decide whether and when a vindictive insider misguidedly gets to somebody's reports in a Cloud benefit. Imitation reports put away in the Cloud, closely the client's

genuine information; likewise fill in as sensors, to identify ill-conceived get to. Once unapproved information get to or introduction is suspected, and later confirmed with test inquiries, for example, we immerse the pernicious insider with false data, keeping in mind the end goal to weaken the client's genuine information. Such preventive assaults that depend on disinformation innovation could give phenomenal levels of security in the Cloud and in informal communities.

REFERENCES

1. *J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," Communications of the ACM,vol. 51, no. 1,pp. 107–113, 2008.*
2. *W. Wang, K. Zhu, L. Ying, J. Tan, and L. Zhang, "Map task scheduling in map reduce with data locality: Throughput and heavy-traffic optimality," in INFOCOM, 2013 Proceedings IEEE.IEEE, 2013, pp. 1609–1617.*
3. *F. Chen,M. Kodialam, and T. Lakshman, "Joint scheduling of processing and shuffle phases in mapreduce systems," in*
4. *INFOCOM,2012 Proceedings IEEE. IEEE, 2012, pp. 1143–1151.*
5. *Y. Wang, W. Wang, C. Ma, and D. Meng, "Zput: A speedy data uploading approach for the hadoop distributed file system," in Cluster Computing (CLUSTER), 2013 IEEE International Conference on. IEEE, 2013, pp. 1–5.*
6. *T. White, Hadoop: the definitive guide: the definitive guide. " O'Reilly Media, Inc.", 2009.*
7. *S. Chen and S. W. Schlosser, "Map-reduce meets wider varieties of applications," Intel Research Pittsburgh, Tech. Rep. IRP-TR-08-05,2008.*
8. *J. Rosen, N. Polyzotis, V. Borkar, Y. Bu, M. J. Carey, M. Weimer,T. Condie, and R. Ramakrishnan, "Iterative mapreduce for large scale machine learning," arXiv preprint arXiv:1303.3517, 2013.*
9. *S. Venkataraman, E. Bodzsar, I. Roy, A. AuYoung, and R. S.Schreiber, "Presto: distributed machine learning and graph processing with sparse matrices," in Proceedings of the 8th ACMEuropean Conference on Computer Systems. ACM,2013, pp. 197–210.*

a. *Matsunaga, M. Tsugawa, and J. Fortes, "Cloudblast: Combining mapreduce and virtualization on distributed resources for bioinformatics applications," in eScience, 2008. eScience'08. IEEE Fourth International Conference on. IEEE, 2008, pp. 222–229.*

10. *J. Wang, D. Crawl, I. Altintas, K. Tzoumas, and V. Markl, "Comparison of distributed data-parallelization patterns for*
11. *big data analysis: A bioinformatics case study," in Proceedings of the Fourth International Workshop on Data Intensive Computing in the Clouds(DataCloud),2013.*
12. *R. Liao, Y. Zhang, J. Guan, and S. Zhou, "Cloudnmf: A mapreduce implementation of nonnegative matrix factorization for largescale biological datasets," Genomics,proteomics &bioinformatics,vol. 12, no. 1, pp. 48–51,2014.*
13. *G. Mackey, S. Sehrish, J. Bent, J. Lopez, S. Habib, and J.Wang,"Introducing map-reduce to high end computing," in*
14. *Petascale Data Storage Workshop, 2008. PDSW'08. 3rd. IEEE, 2008, pp. 1–6.*
15. *W. Yu, G. Xu, Z. Chen, and P. Moulema, "A cloud computing based architecture for cyber security situation awareness," in Communications and Network Security (CNS), 2013 IEEE Conference on. IEEE, 2013, pp. 488–492.*
16. *J. Zhang, H. Zhou, R. Chen, X. Fan, Z. Guo, H. Lin, J. Y. Li,W. Lin, J. Zhou, and L. Zhou, "Optimizing data shufflingin dataparallel computation by understanding user-defined functions," in Proceedings of the 7th Symposium on Networked Systems Design and implementation (NSDI), San Jose, CA, USA, 2012.*

17. *T. Condie, N. Conway, P. Alvaro, J. M. Hellerstein, J. Gerth,J. Talbot, K. Elmeleegy, and R. Sears, "Online aggregation and continuous query support in mapreduce,"in Proceedings of the 2010 ACM SIGMOD International Conference on Management of data.ACM, 2010, pp. 1115–1118.*
18. *T. Condie, N. Conway, P. Alvaro, J. M. Hellerstein, K.Elmeleegy,and R. Sears, Mapreduce online." in NSDI, vol. 10, no. 4, 2010,p. 20.*
19. *J. Lin and C. Dyer, "Data-intensive text processing with mapreduce,"Synthesis Lectures on Human Language Technologies, vol. 3,no. 1, pp. 1–177, 2010.*
20. *P. Costa, A. Donnelly, A. I. Rowstron, and G. O'Shea, "Camdoop:Exploiting in-network aggregation for big data applications." In NSDI, vol. 12, 2012, pp. 3–3.*
21. *On Traffic-Aware Partition and Aggregation in MapReduce for Big Data ApplicationsHuanKe, StudentMember,IEEE, Peng Li, Member, IEEE,SongGuo, Senior Member, IEEE, and MinyiGuo, Senior Member, IEEE*